GPL162004A-507A / GPL162005A-707A
Code Reference Manual

V1.0 - Dec. 31, 2008

3F, No.8, Dusing Rd., Hsinchu Science Park, Hsinchu City 30078, Taiwan, R.O.C.
Tel: 886-3-666-2118 Fax: 886-3-666-2117 Web: www.generalplus.com

Important Note

GENERALPLUS TECHNOLOGY INC. reserves the right to change this documentation without prior notice. Information provided by GENERALPLUS TECHNOLOGY INC. is believed to be accurate and reliable. However, GENERALPLUS TECHNOLOGY INC. makes no warranty for any errors which may appear in this document. Contact GENERALPLUS TECHNOLOGY INC. to obtain the latest version of device specifications before placing your order. No responsibility is assumed by GENERALPLUS TECHNOLOGY INC. for any infringement of patent or other rights of third parties which may result from its use. In addition, GENERALPLUS products are not authorized for use as critical components in life support systems or aviation systems, where a malfunction or failure of the product may reasonably be expected to result in significant injury to the user, without the express written approval of Generalplus.

Table of Content

PAGE

1 START UP CODE FLOW 6

1.1 INTRODUCTION 6

1.2 DETECT PIN DESCRIPTION 7

2 NAND FLASH BOOT FLOW 8

2.1 NAND FLASH BOOT FLOW 8

2.1.1 SLC NAND Flash boot flow 9

2.1.2 MLC NAND Flash boot flow 11

2.2 NAND FLASH BOOT HEADER 12

2.3 NAND FLASH INTRODUCTION 14

2.4 NAND FLASH FUNCTION CODE 14

2.4.1 NAND Flash 0 function 15

2.4.2 NAND Flash 1 function 16

2.4.3 NAND Flash 2 function 17

2.4.4 NAND Flash 3 function 17

2.4.5 NAND Flash 4 function 18

2.4.6 NAND ROM Function Code 19

3 NOR FLASH BOOT FLOW 20

3.1 NOR FLASH BOOT FLOW 20

3.2 NOR FLASH BOOT HEADER 20

3.3 NOR FLASH FUNCTION CODE 21

4 SPI FLASH/ROM BOOT FLOW 22

4.1 SPI BOOT FLOW CHAR 22

4.2 SPI BOOT HEADER 23

4.3 SPI BOOT FUNCTION CODE 25

4.3.1 SPI NOR Flash Function code 25

4.3.2 SPI MCU Function Code 26

4.4 SPI TIMING DIAGRAM 26

5 SDC BOOT FLOW 27

5.1 SDC BOOT FLOW CHAR 27

5.2 SDC BOOT HEADER 28

5.3 SDC FUNCTION CODE 29

6 USB SERVICE LOOP 31

6.1 INTRODUCTION 31

6.2 USB DOWNLOAD FLOW 31

6.3 TOOLS 32

6.4 APPLICATION 34

6.4.1 SPI Flash and SDC boot code update 34

6.4.2 BIOS update or Data write 34

7 ISR VECTOR 35

8 LIBRARY RESOURCE 37

8.1 INTRODUCTION 37

8.2 SPEECH API 38

8.3 CALL BACK FUNCTION 40

Revision History

Revision	Date	By	Remark
1.0	2008/12/31	Jacky Lin	First edit

1 Start up Code flow

1.1 Introduction

After CPU resets or awakes from halt/sleep mode, CPU will first run the start-up code. The beginning address of start-up code is allocated at the reset vector, 0xFFF7, in interrupt table. The flow of GPL162004A-507A/GPL162005A-707A start up code is depicted as follows.

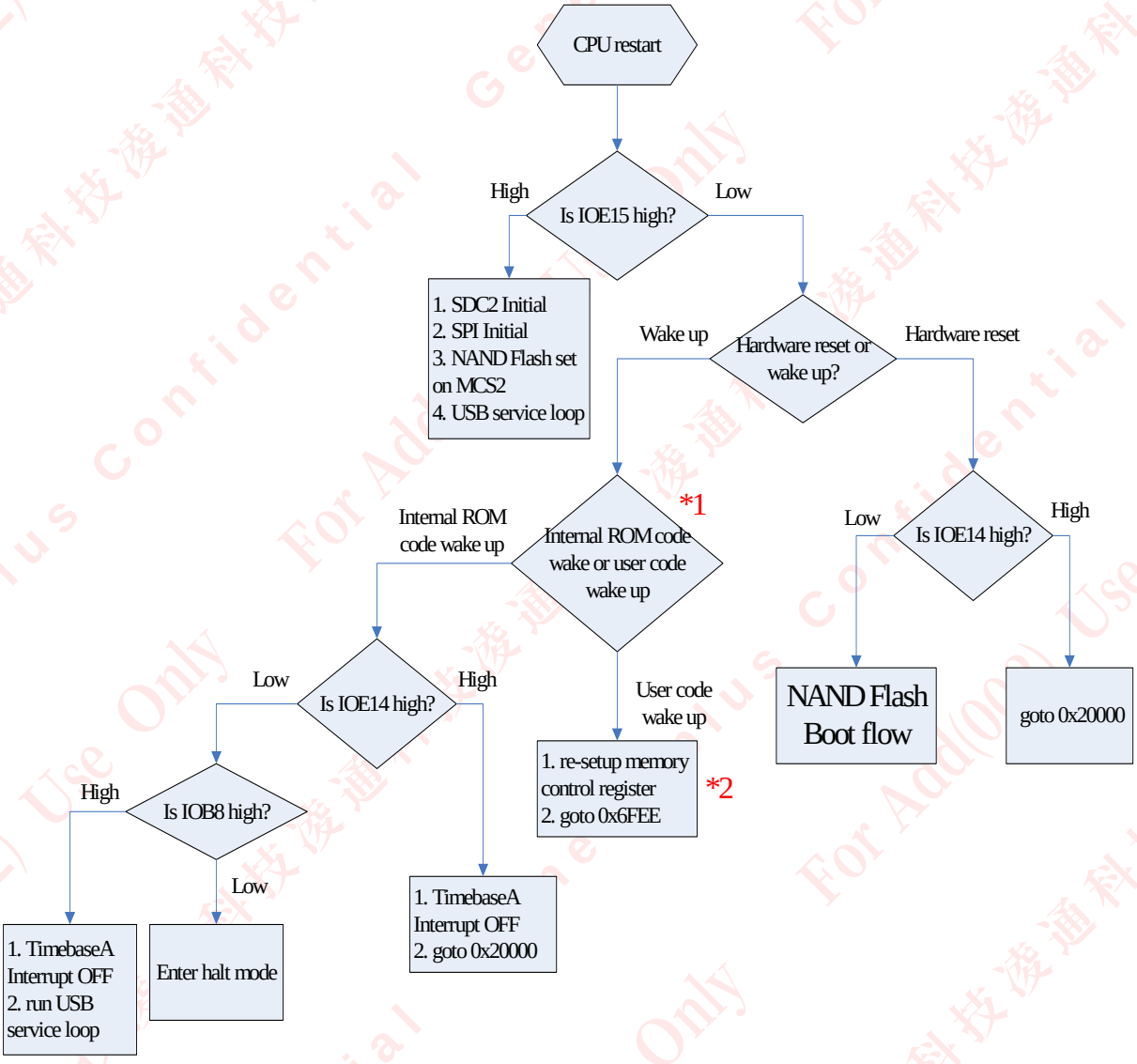


Figure 1-1 start up code flow

Note

*1: In case the boot procedure is not successfully completed even if all internal ROM code boot flows have been executed, CPU will enter halt mode and activate the TimeBaseA 1Hz wake-up source. However, before entering halt mode, CPU will first fill some constant values into the internal RAM, 0x0000~0x0004, which will be used to signal CPU that it is an internal ROM code halt mode when it

awakes. Thus, suppose GPL162004A-507A/GPL162005A-707A ROM code boot flow is executed completely and user's code dominates the system. A "0" should be written in any location inside internal RAM 0x0000 ~ 0x 0004 before entering halt mode that guarantees user's code wake-up procedure will be executed after CPU awakes.

*2: What "user code wake-up" means that after the entire boot process is completed, application program enters halt mode and then wakes up. Because the memory control registers will be reset as default after CPU resets, memory control registers should be re-configured before CPU executes user wake-up code in order to assure that program is executed correctly. In GPL162004A-507A/GPL162005A-707A ROM code, CPU will copy the memory setting (stored at 0x0005~0x0009 inside internal RAM) to memory control registers (MCS0~MCS4) in sequences.

1.2 Detect Pin Description

Pin number	Function description
IOB8	This signal is to check if USB cable is plugged in or not after wake up from halt mode of internal ROM code. IOB8 will be in high state if USB is inserted. In application circuit, USB power need wired to IOB8 through a 68Kohm resistor by user.
IOE14	This signal is to detect external memory card insertion. IOE14 will be in high state if an external memory card is inserted. In application circuit, when a card is inserted, IOE14 is high; when a card is removed, IOE14 goes low.
IOE15	This signal is to check if USB download function is enabled. When CPU reset, CPU will execute USB service loop function if IOE15 is detected high. This option allows users to implement memory update or download function in the applications.

Table1 detect pin function

2 NAND Flash Boot Flow

2.1 NAND Flash Boot Flow

When IOE15 and IOE14 are detected low, CPU will execute NAND Flash boot flow. In this boot flow, CPU will first read NAND ID to determine whether ID is supported. If yes, CPU starts moving NAND data and checks the tag to see if the memory boot code is existed or damaged. If the tag is incorrect or boot code is damaged, CPU will check NOR flash boot flow. In NAND Flash boot mode, it recognizes SLC or MLC NAND flash according NAND Flash ID since these two NAND flash uses different ECC detections and corrections. In SLC NAND Flash, it uses 2-bit detection and 1-bit correction ECC; on the other hand, MLC NAND Flash uses 8-bit detection and correction ECC. Please refer to Fig. 2-1 and Fig. 2-2 flowchart. If NAND ID check is not in the supported list in the first place, IOD14 status will be the key flag for determination. If IOD14 is high, it means NAND ROM memory is applied and NAND ROM boot flow will be executed. The NAND Flash boot flow is as follows.

2.1.1 SLC NAND Flash boot flow

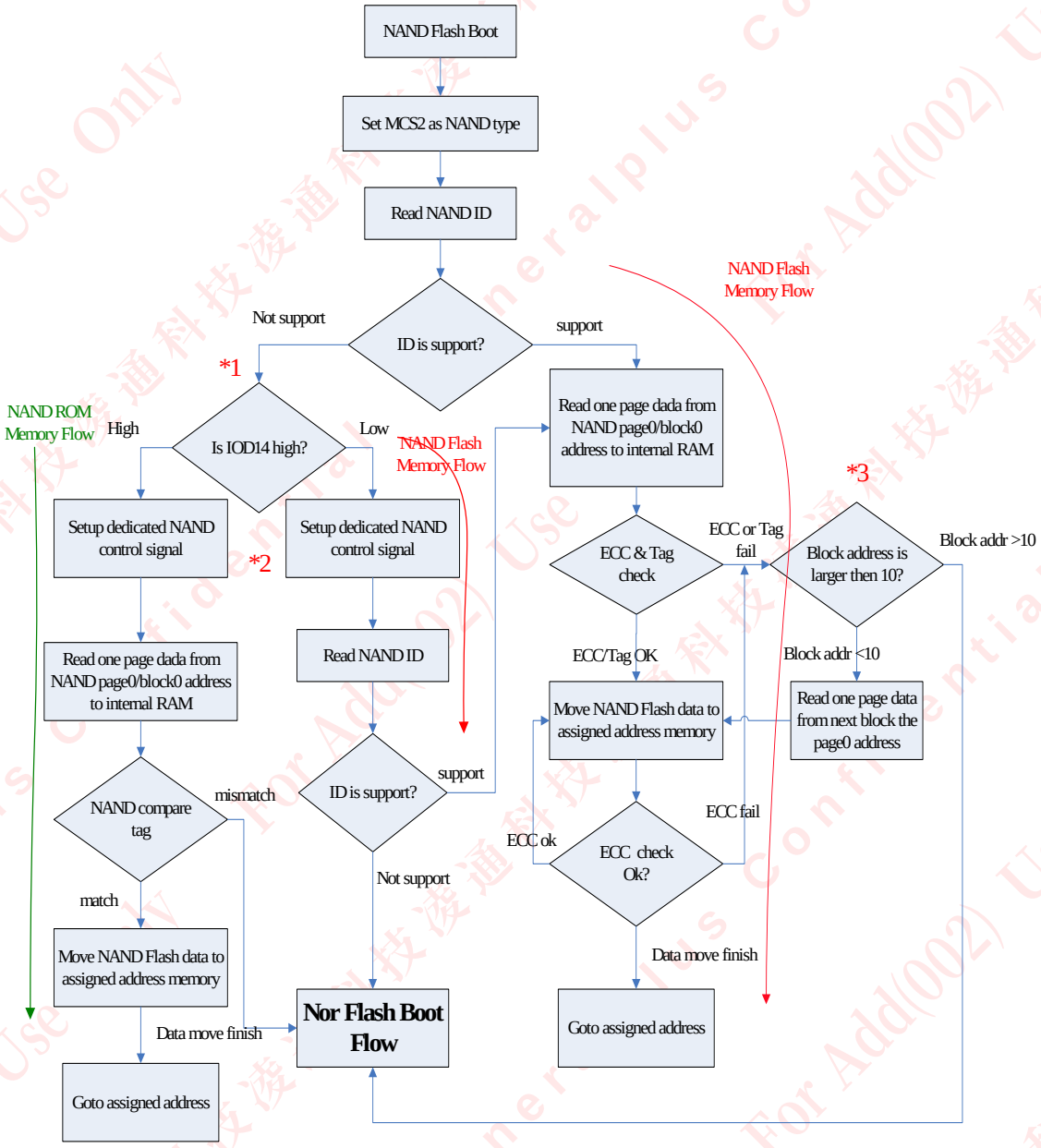


Figure 2-1 SLC NAND Flash boot flow

Note:

*1. If a NAND ID is not in our internal ROM supporting list, CPU will starts detecting the status of IOD14 of which default is XA22. Therefore, CPU will first disable XA22 function before detect IOD14. When IOD14 is low, meaning no NAND ROM type memory is connected. At this point, CPU will configure additional NAND Flash interface and start read NAND Flash ID. When IOD14 is high, meaning NAND ROM memory type is connected. Then, CPU will configure additional NAND Flash interface and read NAND ROM data.

*2. Because of the timing differences between NAND ROM and ordinary NAND flash, we have to use additional NAND Flash control pin to connect with it when NAND ROM is applied. As a result, hardware configuration must be met as the following table indicates when NAND ROM type memory is used. For more information, please refer to GPL162004A/GPL162005A programming guide.

	Normal NAND Flash Memory	NAND ROM Memory (dedicated NAND Flash control signal)
NAND[D7..D0]	Connect CPU XA8~XA15	Connect CPU XA8~XA15
NAND_ALE	Connect CPU IOD8 (XA16)	Connect CPU IOD13 (XA21)
NAND_CLE	Connect CPU IOD9 (XA17)	Connect CPU IOB4
NAND_WE	Connect CPU MWE	Connect CPU IOD14 (XA22)
NAND_OE	Connect CPU MOE	Connect CPU IOD15 (XA23)
NAND_RDY	Connect CPU IOC12 (NAND RDY)	Connect CPU IOC12 (NAND RDY)
NAND_CS	Connect CPU IOD2 (MCS2)	Connect CPU IOD2 (MCS2)

Note: NAND ROM memory interface connects normal NAND Flash memory and works well.

*3. NAND Flash boot first moves and checks data from block0/page0. When check fails, it continues moving and checking data from the next block's page 0. To increase the success rate of NAND flash boot, we recommend backing up the boot code at the first few blocks of NAND Flash.

2.1.2 MLC NAND Flash boot flow

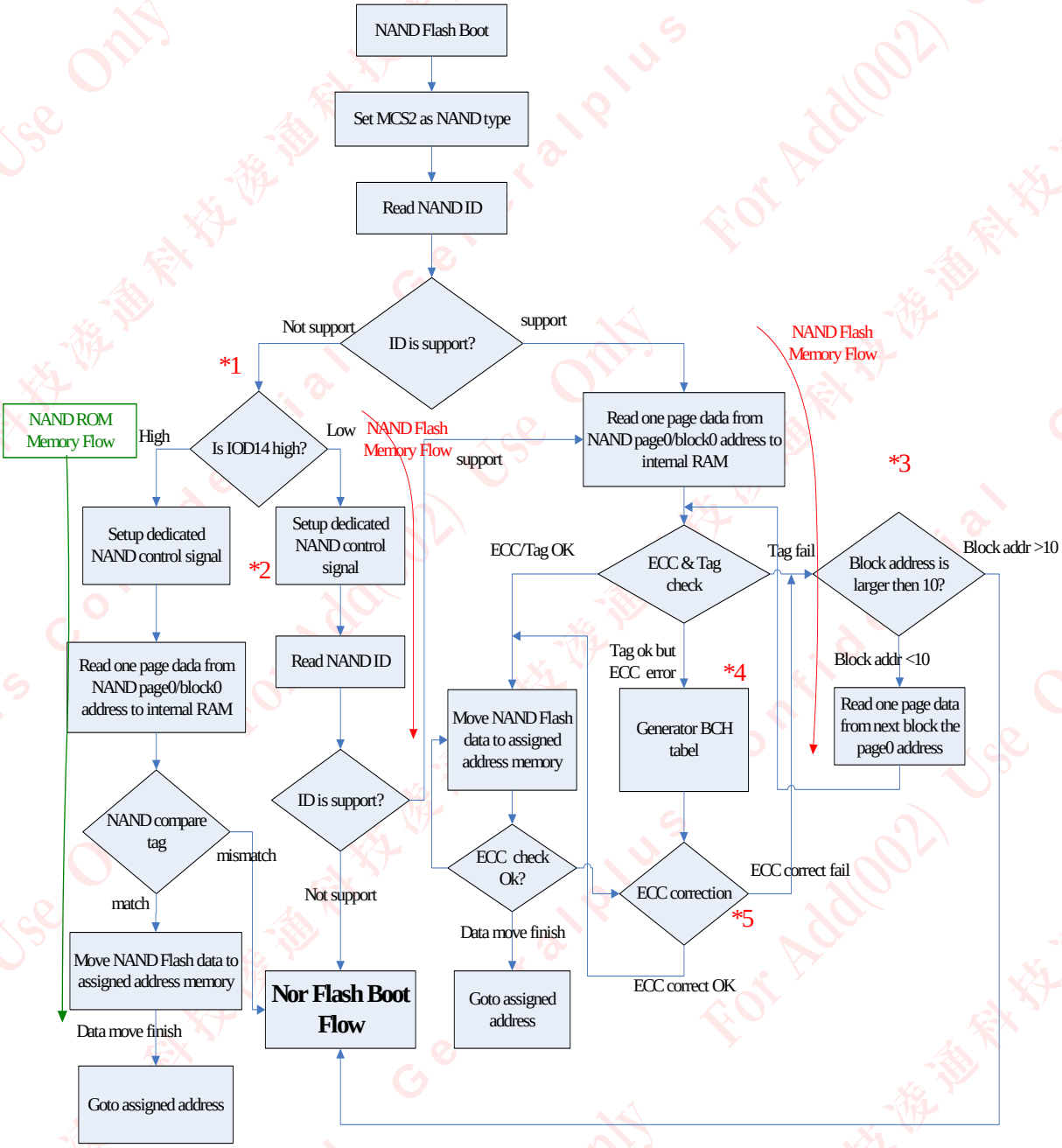


Figure 2-2 MLC NAND Flash boot flow

Note:

*1. please refer to section 2.1.1 note1.

*2. please refer to section 2.1.1 note 2.

*3. please refer to section 2.1.1 note 3.

*4. The 8-bit ECC correct code needs a16K-word table (BCH table). About ROM size, the BCH table will be generated by internal ROM code. It is only generated one time in entire MLC NAND Flash boot flow. Both the generation and BCH table location are determined by internal ROM code. If RAM is not externally added or if the start address of external added RAM is over 0x400000, internal ROM code will not perform the ECC correction.

*5. ECC correction code will need some global variables which will be allocated at the internal RAM 0x2890 ~ 0x2BE0 (total 0x351 words). Therefore, during MLC NAND Flash boot sequence, DO NOT assign the boot code to this address that will cause ECC correction failed.

2.2 NAND Flash Boot Header

1	2	3	4	5	6	7	8
Tag1	Tag2	Tag3	Tag4	Tag5	CS0 setup	CS1 setup	CS2 setup
9	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CS3 setup	CS4 setup	DesAddrL	DesAddrH	Sector count	PSRAM Ctrl	Write/Read Timing	0
17	**18**	**19**	**20**	**21**	**22**	**23**	**24**
SDRAM Enable	SDRAM CBRCYC	SDRAM Timing	SDRAM Ctrl1	SDRAM Misc	SDRAM Ctrl0	BCH Table AddrL	BCH Table AddrLH
25	**26**	**27**	**28**	**29**	**30**	**31**	**32**
SouAddrL	SouAddrH	Sys CLK	PLL setup	SPI CLK	0	0	0

NAND Flash boot header includes 32 words that need to be arranged in the beginning of user's code, where is in the address 0 of NAND Flash page0/block0. User's code starts from 33rd word. The definition of each word in the header is as follows,

Tag1: 0x4750 for NAND boot

Tag2: 0x6E61 for NAND boot

Tag3: 0x6E64 for NAND boot

Tag4: 0x6E61 for NAND boot

Tag5: 0x6E64 for NAND boot

CS0 setup: the value for MCS0 control register. CPU will fill this value to the register 0x7820 after Tag1~Tag5 are checked correctly.

CS1 setup: the value for MCS1 control register. CPU will fill this value to the register 0x7821 after Tag1~Tag5 are checked correctly.

CS2 setup: the value for MCS2 control register. CPU will fill this value to the register 0x7822 after Tag1~Tag5 are checked correctly.

CS3 setup: the value for MCS3 control register. CPU will fill this value to the register 0x7823 after Tag1~Tag5 are checked correctly.

CS4 setup: the value for MCS4 control register. CPU will fill this value to the register 0x7824 after Tag1~Tag5 are checked correctly.

DesAddrL, DesAddrH: the designated beginning memory address to store data from NAND Flash page0/block0. The memory offset from this address cannot be over 4M words in the first bank. It means that data copy from NAND Flash can only be stored in first 4MW of memory mapping in GPL162004A/GPL162005A.

Sector count: the designated data count to be read from NAND Flash. The unit for the data is based on sector (512 Bytes) for SLC NAND. The unit for the data is based on Page (2K or 4K Bytes) for MLC NAND.

PSRAM Ctrl: the value for P_PSRAM_Ctrl control register. CPU will fill this value to the register 0x7825 after Tag1~Tag5 are checked correctly.

Write/Read Timing: the value for P_RAW_WAR control register. CPU will fill this value to the register 0x782D after Tag1~Tag5 are checked correctly.

SDRAM Enable: the value for the control register which enables SDRAM. CPU will fill this value to the register 0x782F after Tag1~Tag5 are checked correctly.

SDRAM CBRCYC: the value for the control register which refreshes SDRAM. CPU will fill this value to the register 0x783D after Tag1~Tag5 are checked correctly.

SDRAM Timing: the value for SDRAM timing control register. CPU will fill this value to the register 0x783C, after Tag1~Tag5 are checked correctly.

SDRAM Ctrl1: the value for the control register which initializes SDRAM. CPU will fill this value to the register 0x783B after Tag1~Tag5 are checked correctly.

SDRAM Misc: the value for the control register which SDRAM timing adjust. CPU will fill this value to the register 0x783E after Tag1~Tag5 are checked correctly.

SDRAM Ctrl0: the value for the control register which initializes SDRAM. CPU will fill this value to the register 0x783A after Tag1~Tag5 are checked correctly.

BCH Table AddrL, BCH Table AddrH: when the **DesAddrH** parameter MSB is 1, then BCH table address will be assigned via **BCH Table AddrL, BCH Table AddrH** parameter**.**

SouAddrL, SouAddrH: these two parameters are valid only on SPI MCU boot mode.

Sys CLK: N/A in this boot mode. User can set to 0.

PLL setup: N/A in this boot mode. User can set to 0.

SPI CLK: N/A in this boot mode. User can set to 0.

2.3 NAND Flash Introduction

Once GPL162004A-507A/GPL162005A-707A enters NAND Flash boot function, CPU will read NAND Flash data and next, copy them into internal RAM first. Generalplus groups NAND Flash into five categories which cover as many NAND Flash types as possible on the market today. By checking the NAND Flash ID, it determines which NAND Flash function code will be downloaded to internal RAM and be executed.

NFType	Size	Address Cycle	CLE30 command	Page Size	Block Size	Column Address	Row Address
NAND0	64M~128Mx8	4	N	512 + 16	32 page	1	3
NAND1	128Mx8	4	Y	2048 + 64	64 page	2	2
NAND2	256M~32G x8	5	Y	2048 + 64	64 page	2	3
NAND3	256M~32G x8	5	Y	2048 + 64	128 page	2	3
NAND4	256M~64G x8	5	Y	4096+128	128 page	2	3

Table2 NAND Flash type definition

Note:

1. GPL162004A-507A/GPL162005A-707A does not support for those NAND Flash devices' sizes under 64MB in NAND Flash boot function.
2. To execute NAND Flash boot function using GPL162004A-507A/GPL162005A-707A internal ROM code, it must connect NAND Flash to MCS2.
3. NAND3 and NAND4 are MLC NAND Flash type.

2.4 NAND Flash Function Code

The following table lists the most commonly used NAND Flash devices supported by GPL162004A-507A/GPL162005A-707A ROM code.

Manufacture	ID Number	Size	NAND Flash Function Call
Samsung	0xEC, 0x76	64MB	Call NAND0
	0xEC, 0x79	128MB	Call NAND0
	0xEC, 0xF1	128MB	Call NAND1
	0xEC, 0xDA	256MB	Call NAND2 or NAND3
	0xEC, 0xDC	512MB	Call NAND2 or NAND3
	0xEC, 0xD3	1GB	Call NAND2 or NAND3
	0xEC, 0xD5	2GB	Call NAND2 or NAND3
Hynix	0xAD, 0x76	64MB	Call NAND0
	0xAD, 0x79	128MB	Call NAND0
	0xAD, 0xF1	128MB	Call NAND1
	0xAD, 0xDA	256MB	Call NAND2 or NAND3
	0xAD, 0xDC	512MB	Call NAND2 or NAND3

Manufacture	ID Number	Size	NAND Flash Function Call
	0xAD, 0xD3	1GB	Call NAND2 or NAND3
	0xAD, 0xD5	2GB	Call NAND2 or NAND3
Toshiba	0x98, 0x76	64MB	Call NAND0
	0x98, 0x79	128MB	Call NAND0
	0x98, 0xF1	128MB	Call NAND1
	0x98, 0xDA	256MB	Call NAND2 or NAND3
	0x98, 0xDC	512MB	Call NAND2 or NAND3
	0x98, 0xD3	1GB	Call NAND2 or NAND3
	0x98, 0xD5	2GB	Call NAND2 or NAND3
ST	0x20, 0x76	64MB	Call NAND0
	0x20, 0x79	128MB	Call NAND0
	0x20, 0xF1	128MB	Call NAND1
	0x20, 0xDA	256MB	Call NAND2
	0x20, 0xDC	512MB	Call NAND2
	0x20, 0xD3	1GB	Call NAND2
Micron	0x2C, 0xDA	256MB	Call NAND2
	0x2C, 0xDC	512MB	Call NAND2
	0x2C, 0xD3	1GB	Call NAND2
	0x2C, 0xD5	2GB	Call NAND2

Table3 Supported NAND Flash ID

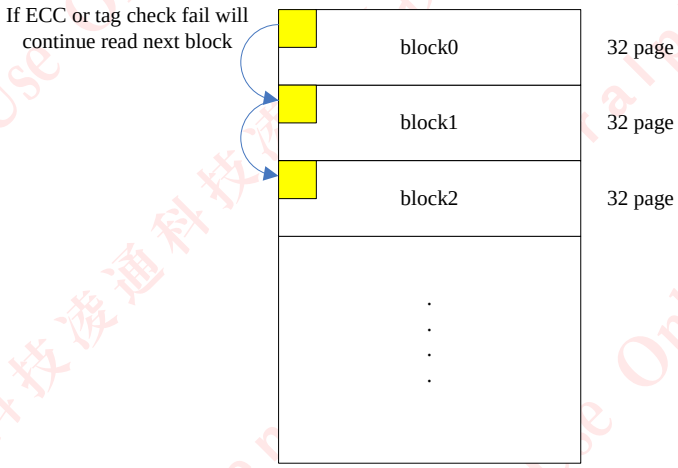
Note: The NAND Flash commands corresponding to NAND Flash devices in GPL162004A-507A/GPL162005A-707A ROM code are based on the NAND Flash ID number read by CPU. Some Flash manufacturers, however, define various NAND Flash type devices with a same ID; as a consequence, some of devices are supported in GPL162004A-507A/GPL162005A-707A, but some are not in this case. For those unsupported devices, users can configure them via E-Fuse control register (0x7AE0 [b2:b0]) to make the devices functioning.

2.4.1 NAND Flash 0 function

In NAND Flash 0 function, it will read one page data form NAND Flash page0 in bank0, where one page is 528 bytes. It reads 512 bytes data at each page and reads 513th ~518th bytes for ECC check. Every two bytes of data read from NAND0 type Flash will be combined as a word and will be stored from 0x3000 of internal RAM in sequences. CPU will read first page data from NAND Flash and then perform header data check. Once the first 5-word tag is verified successfully, the following parameters, MCS0~MCS4 and SDRAM, will be copied to corresponding control registers and then start copying data from page0/block0 address 0 of NAND Flash to the designated address. After transfer is completed, CPU will jump to the address, where is the designated address plus 0x20 of offset, and then execute the program.

If ECC check error or tag check failure occurs during NAND Flash transfer, CPU will continue reading next block page0/address0 data and performing ECC and tag check. If 8 successive blocks are all

check failed, CPU will got o next boot mode that is NOR Flash boot. So programmer can assign some blocks to backup boot code. This will reduce CPU boot failure due to boot code destroyed at NAND Flash operation.

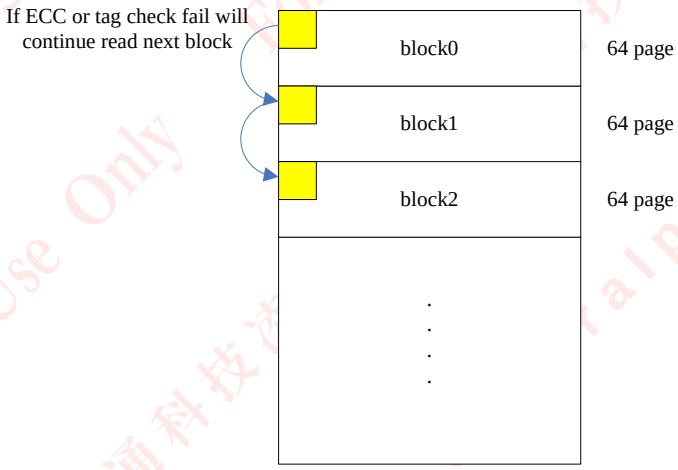


Note: In writing procedure of NAND0 type Flash boot code, the ECC result of each page NAND Flash data (512 bytes) should be attached in 512th~517th bytes of the page.

2.4.2 NAND Flash 1 function

In NAND Flash 1 function, it will read one page data form NAND Flash page0 in bank0, where one page is 2112 bytes. It reads 2048 bytes data at each page and reads 2049th ~2112th bytes for ECC check. Every 16 of these 64 bytes is taken as a unit for 512-byte ECC result of 2048 bytes data and then only first 6 bytes of a unit are used to store ECC value. Every two bytes of data read from NAND1 type Flash will be combined as a word and will be stored from 0x3000 of internal RAM sequentially. CPU will read the first page data from NAND Flash and then perform header data check. Once the first 5-word tag is verified successfully, the following parameters, MCS0~MCS4 and SDRAM, will be copied to corresponding control registers and then start copying data from page0/block0 address 0 of NAND Flash to the designated address. After transfer is done, CPU will jump to the address, where is the designated address plus 0x20 of offset, and then execute the program.

If ECC check error or tag check failure occurs during NAND Flash transferring, CPU will continue reading next block page0/address0 data and doing ECC and tag check. If 10 successive blocks are all check failed, CPU will go to next boot mode that is NOR Flash boot. Thus, programmer can assign some blocks to backup boot code. This will reduce CPU boot failure due to boot code destroyed at NAND Flash operation.



Note: In writing procedure of NAND1 type Flash boot code, the ECC result of each page NAND Flash data (2048 bytes) should be attached in 2049th~2102nd bytes of the page which is similar to read procedure depicted above.

2.4.3 NAND Flash 2 function

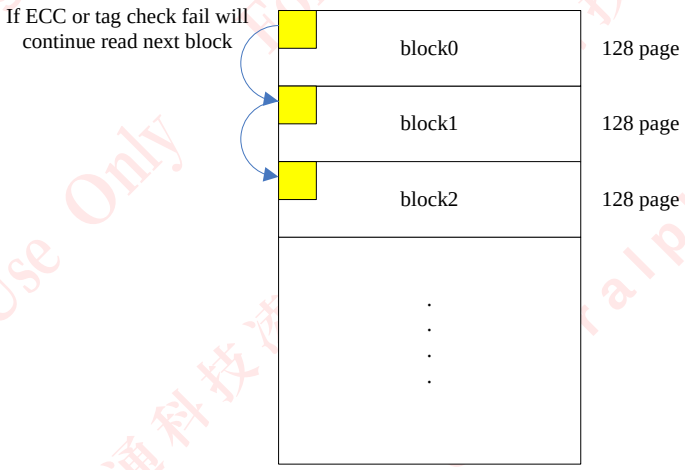
Except for address command cycle described at Table 2, NAND Flash 2 function is almost the same as NAND Flash 1 function. Please refer to section 2.4.2 for more details.

2.4.4 NAND Flash 3 function

If NAND ID type is a 128-page NAND flash or above, CPU assumes that the NAND Flash is possibly the MLC NAND Flash and therefore, CPU will read one page at a time from NAND Flash and execute BCH 8 bit ECC check.

In NAND Flash 3 function, it will read one page data form NAND Flash page0 in bank0, where one page is 2112 bytes. Every two bytes of data read from NAND3 type Flash will be combined as a word and stored from 0x1000 of internal RAM sequentially. CPU will read first page data from NAND Flash and then perform header data check. Once the first 5-word tag is verified successfully, the following parameters, MCS0~MCS4 and SDRAM, will be copied to corresponding control registers and then start copying data from page0/block0 address 0 of NAND Flash to the designated address. After transfer is done, CPU will jump to the address, where the designated address plus 0x20 of offset, and then execute the program.

If ECC check error or tag check failure occurs during NAND Flash transferring, CPU will continue reading next block page0/address0 data and doing ECC and tag check. If 10 successive blocks are all check failed, CPU will go to next boot mode that is NOR Flash boot. Programmer can assign some blocks to backup boot code. This reduces CPU boot failuredue to boot code destroyed at NAND Flash operation.



Note1: In this mode, CPU will only move 2048 byte of data to the designated memory every time and dump the rest of 64 bytes.

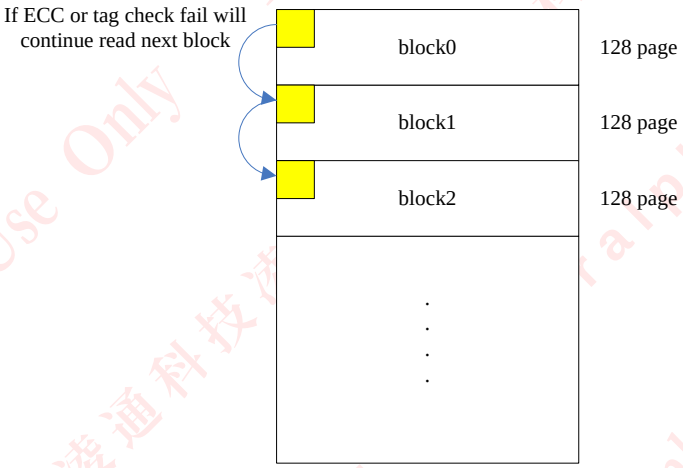
Note2: MLC NAND Flash boot will do ECC correction when ECC check failed. The ECC correct code will use global variable that is located at internal RAM 0x2890 ~ 0x2BE0 and therefore, users don't assigned boot code in this area or, ECC correction will be failed.

2.4.5 NAND Flash 4 function

If NAND ID type is a 128-page NAND flash or above, CPU assumes that the NAND Flash is possibly the MLC NAND Flash and therefore, CPU will read one page at a time from NAND Flash and execute BCH 8 bit ECC check.

In NAND Flash 4 function, it will read one page data form NAND Flash page0 in bank0, where one page is 4224 bytes. Every two bytes of data read from NAND4 type Flash will be combined as a word and stored from 0x1000 of internal RAM sequentially. CPU will read first page data from NAND Flash and then perform header data check. Once the first 5-word tag is verified successfully, the following parameters, MCS0~MCS4 and SDRAM, will be copied to corresponding control registers and then start copying data from page0/block0 address 0 of NAND Flash to the designated address. After transfer is done, CPU will jump to the address, where is the designated address plus 0x20 of offset, and then execute the program.

If ECC check error or tag check failure occurs during NAND Flash transferring, CPU will continue reading next block page0/address0 data and doing ECC and tag check. If 10 successive blocks are all check failed, CPU will go to next boot mode that is NOR Flash boot. Programmer can assign some blocks to backup boot code. This can reduce CPU boot failure due to boot code destroyed at NAND Flash operation.



Note1: In this mode, CPU will only move 4096 byte of data to the designated memory every time and dump the rest of 128 bytes.

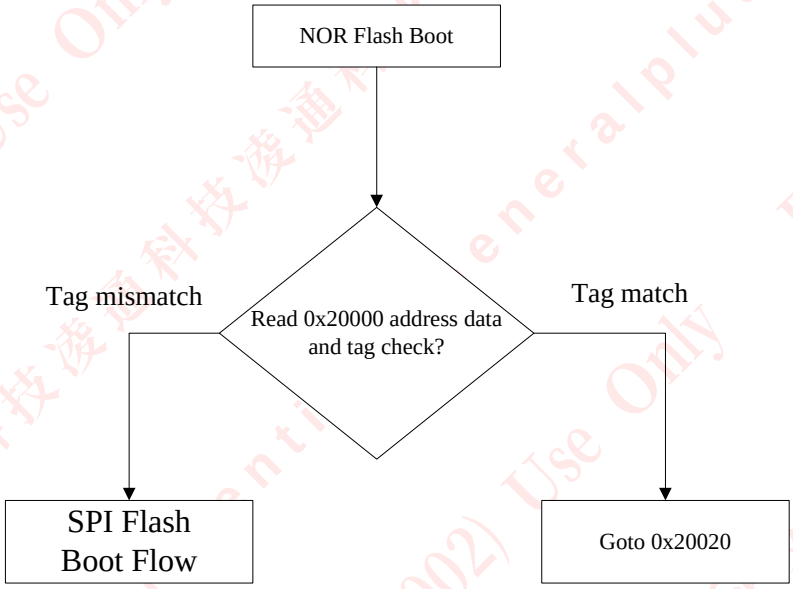
Note2: MLC NAND Flash boot will do ECC correction when ECC check failed. The ECC correct code will use global variable that is located at internal RAM 0x2890 ~ 0x2BE0. So users don't assigned boot code in this area or the ECC correction will be failed.

2.4.6 NAND ROM Function Code

If a NAND ID is not in our internal ROM supporting list and IOD14 is pulled high by resistor, CPU will execute NAND ROM boot flow. In NAND ROM function, it will read one page data form NAND ROM page0 in bank0, where one page is 512 bytes. Every two bytes of data read from NAND ROM will be combined as a word and will be stored from 0x3000 of internal RAM sequentially. CPU will read the first page data from NAND ROM and then perform header data check. Once the first 5-word tag is verified successfully, the parameters, MCS0~MCS4 and SDRAM, will be copied to corresponding control registers and then start copying data from page0/block0 address 0 of NAND ROM to the designated address. After transfer is done, CPU will jump to the address, where is the designated address plus 0x20 of offset, and then execute the program. In other words, if a user wants to use NAND ROM boot configuration, the first 32 words of NAND ROM boot code should be reserved and first 5 words of them should be in "GPnandnand"; the user's program begins from 33rd word. In NAND ROM mode, it will not perform ECC check while moving data from NAND ROM. This assures the data in NAND ROM is correct.

3 NOR Flash Boot Flow

3.1 NOR Flash Boot Flow



3.2 Nor Flash Boot Header

1	2	3	4	5	6	7	8
Tag1	Tag2	Tag3	Tag4	Tag5	CS0 setup	CS1 setup	CS2 setup
9	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CS3 setup	CS4 setup	DesAddrL	DesAddrH	Sector count	PSRAM Ctrl	Write/Read Timing	0
17	**18**	**19**	**20**	**21**	**22**	**23**	**24**
SDRAM Enable	SDRAM CBRCYC	SDRAM Timing	SDRAM Ctrl1	SDRAM Misc	SDRAM Ctrl0	BCH Table AddrL	BCH Table AddrLH
25	**26**	**27**	**28**	**29**	**30**	**31**	**32**
SouAddrL	SouAddrH	Sys CLK	PLL setup	SPI CLK	0	0	0

NOR Flash boot header includes 32 words which need to be arranged at the beginning of NOR Flash in MCS0, 0x20000. User's code starts from 33rd word. The definition of each word of the header is as follows:

Tag1: 0x4750 for NOR Flash boot

Tag2: 0X6E6F for NOR Flash boot

Tag3: 0X726E for NOR Flash boot

Tag4: 0X6F72 for NOR Flash boot

Tag5: 0X6E6F for NOR Flash boot

CS0 setup: N/A in this boot mode

CS1 setup: N/A in this boot mode

CS2 setup: N/A in this boot mode

CS3 setup: N/A in this boot mode

CS4 setup: N/A in this boot mode

DesAddrL, DesAddrH: N/A in this boot mode

Sector count: N/A in this boot mode

PSRAM Ctrl: N/A in this boot mode

Write/Read Timing: N/A in this boot mode

SDRAM Enable: N/A in this boot mode

SDRAM CBRCYC: N/A in this boot mode

SDRAM Timing: N/A in this boot mode

SDRAM Ctrl1: N/A in this boot mode

SDRAM Misc: N/A in this boot mode.

SDRAM Ctrl0: N/A in this boot mode

BCH Table AddrL, BCH Table AddrH: N/A in this boot mode.

SouAddrL, SouAddrH: N/A in this boot mode

Sys CLK: N/A in this boot mode

PLL setup: N/A in this boot mode

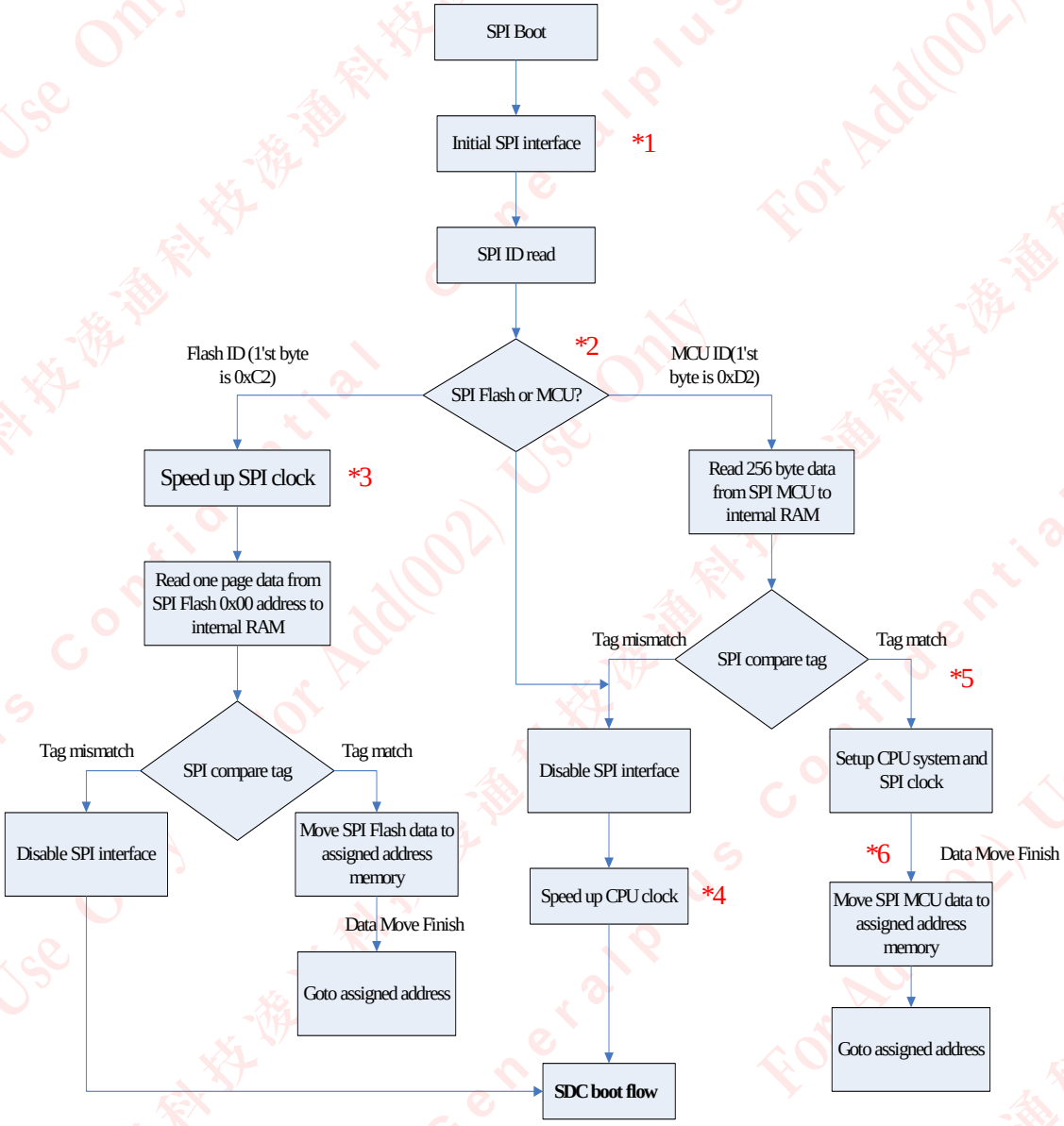
SPI CLK: N/A in this boot mode

3.3 NOR Flash Function Code

After NAND Flash boot code fails, CPU will check NOR Flash boot code immediately. In this boot mode, Generalplus recommends that NOR Flash should be connected to MCS0, which the start address is 0x20000. Therefore, when CPU jumps to NOR Flash boot function, CPU will read data from 0x20000 and then do NOR Flash tag comparison. The NOR Flash tag is “0x4750”, “0x6E6F, 0x726E, 0x6F72, 0x6E6F” (“GPnornorno”). If the comparison is matched, CPU will go to address 0x20020 to execute the program; otherwise, CPU will call SPI Flash boot function instead once comparison fails. In other words, if a user wants to use NOR Flash boot configuration, the first 32 words of Nor Flash boot code should be reserved and first 5 words of them should be in ”GPnornorno” and the user’s program begins from 33rd word.

4 SPI Flash/ROM Boot Flow

4.1 SPI Boot Flow Char



Note*1: SPI initial will set the IOB[7..4] to SPI interface, SPI clock to 48KHz, and CPU clock to 6MHz. Please refer to 4.4 SPI timing section for more information on SPI timing picture.

Note*2: SPI communication command format, please reference Generalplus GPR25LXXX series datasheet for SPI communication command format details. If the 1st byte of SPI ID is 0xD2, it means SPI device is an MCU and therefore, the 2nd and 3rd byte of the ID represent the SPI MCU address for storing SPI header data. After that, GPL162004A-507A/GPL162005A-707A will fetch SPI MCU header data from that address and move data to designation address based on the header data information.

SPI device	ID number		
	1'st byte	2'nd byte	3'rd byte
SPI Flash	0xC2 (manufacture)	0x20 (memory type)	0xXX (memory size)
SPI MCU	0xD2 (manufacture)	Header data high address	Header data low address

Note*3: If SPI device is Flash memory, SPI clock is set to 12MHz and CPU clock to 48MHz.

Note*4: If SPI MCU tag check fails, SPI interface is disabled and clock is set to 48MHz.

Note*5: If SPI MCU tag comparison matches, CPU will get 27th~29th words parameter of SPI boot header and use these parameters to set up system clock and SPI clock. The 27th word parameter sets up the system clock selection, valid only bit0~bit2; other bits must be cleared "0". The 28th word parameter sets PLL clock, valid bit0~bit5; other bits must be cleared "0". The 29th word parameter is for SPI clock, valid bit0~bit2; other bits must be cleared "0". The system clock, PLL clock and SPI clock control register locate as CPU address 0x7807, 0x7817 and 0x7940. For more information about descriptions of these control registers, please refer to GPL162004A/GPL162005A programming guide.

Note*6: If SPI MCU tag comparison matches, SPI MCU stars moving data; SPI MCU data address is defined in the 25th and 26th bytes of header data.

4.2 SPI Boot Header

1	2	3	4	5	6	7	8
Tag1	Tag2	Tag3	Tag4	Tag5	CS0 setup	CS1 setup	CS2 setup
9	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CS3 setup	CS4 setup	DesAddrL	DesAddrH	Sector count	PSRAM Ctrl	Write/Read Timing	0
17	**18**	**19**	**20**	**21**	**22**	**23**	**24**
SDRAM Enable	SDRAM CBRCYC	SDRAM Timing	SDRAM Ctrl1	SDRAM Misc	SDRAM Ctrl0	BCH Table AddrL	BCH Table AddrLH
25	**26**	**27**	**28**	**29**	**30**	**31**	**32**
SouAddrL	SouAddrH	Sys CLK	PLL setup	SPI CLK	0	0	0

SPI NOR Flash boot header includes 32 words which need to be arranged at the beginning of SPI NOR Flash, 0x000000. User's code starts from 33rd word. The definition of each word of the header is as follows,

Tag1: 0x4750 for SPI NOR Flash boot

Tag2: 0X7370 for SPI NOR Flash boot

Tag3: 0x6973 for SPI NOR Flash boot

Tag4: 0x7069 for SPI NOR Flash boot

Tag5: 0x7370 for SPI NOR Flash boot

CS0 setup: the value for MCS0 control register. CPU will fill this value to the register, 0x7820, after Tag1~Tag5 are checked correctly.

CS1 setup: the value for MCS1 control register. CPU will fill this value to the register, 0x7821, after Tag1~Tag5 are checked correctly.

CS2 setup: the value for MCS2 control register. CPU will fill this value to the register, 0x7822, after Tag1~Tag5 are checked correctly.

CS3 setup: the value for MCS3 control register. CPU will fill this value to the register, 0x7823, after Tag1~Tag5 are checked correctly.

CS4 setup: the value for MCS4 control register. CPU will fill this value to the register, 0x7824, after Tag1~Tag5 are checked correctly.

DesAddrL, DesAddrH: the designated beginning memory address to store data from NAND Flash page0/block0. The memory offset from this address cannot be over 4M words of the first bank. It means data copied from SPI NOR Flash can only be stored in first 4MW of memory mapping of GPL162004A/2005A.

Sector count: the designated data count to be read from SPI NOR Flash. The unit for the data is based on sector (512 Bytes).

PSRAM Ctrl: the value for P_PSRAM_Ctrl control register. CPU will fill this value to the register, 0x7825, after Tag1~Tag5 are checked correctly.

Write/Read Timing: the value for P_RAW_WAR control register. CPU will fill this value to the register, 0x782D, after Tag1~Tag5 are checked correctly.

SDRAM Enable: the value for the control register to enable SDRAM. CPU will fill this value to the register, 0x782F, after Tag1~Tag5 are checked correctly.

SDRAM CBRCYC: the value for the control register to refresh SDRAM. CPU will fill this value to the register, 0x783D, after Tag1~Tag5 are checked correctly.

SDRAM Timing: the value for SDRAM timing control register. CPU will fill this value to the register, 0x783C, after Tag1~Tag5 are checked correctly.

SDRAM Ctrl1: the value for the control register to initialize SDRAM. CPU will fill this value to the register, 0x783B, after Tag1~Tag5 are checked correctly.

SDRAM Misc: the value for the control register which SDRAM timing adjust. CPU will fill this value to the register, 0x783E, after Tag1~Tag5 are checked correctly.

SDRAM Ctrl0: the value for the control register to initialize SDRAM. CPU will fill this value to the register, 0x783A, after Tag1~Tag5 are checked correctly.

BCH Table AddrL, BCH Table AddrH: N/A in this boot mode.

SouAddrL, SouAddrH: In SPI MCU boot, these two bytes represent the starting address of data movement for SPI MCU.

Sys CLK: this parameter is valid only in SPI MCU boot mode.

PLL setup: this parameter is valid only in SPI MCU boot mode.

SPI CLK: this parameter is valid only in SPI MCU boot mode.

4.3 SPI boot Function Code

Once NOR Flash boot mode check fails, CPU will execute SPI boot flow. In order to connect with lower speed of SPI device, e. g. a low speed device using GPIO for SPI interface, GPL162004A-507A/GPL162005A-707A will set the CPU clock to 6MHz and SPI clock to 48KHz. First, read the SPI ID to identify where SPI device is Flash or MCU. If the 1st byte of ID is 0xC2, SPI NOR Flash boot flow is executed. If the 1st byte of ID is 0xD2, SPI MCU boot flow will be executed instead. Suppose the 1st byte of ID is neither these two values (0xC2, 0xD2), SPI boot function will not be executed and SPI interface will be disabled. The IOB[7..4] will return to default status. Therefore, CPU clock is set to 48MHz and continues to execute SD card boot function.

4.3.1 SPI NOR Flash Function code

If the 1st byte of ID is 0xC2, it means the type of SPI device is NOR Flash; CPU clock is set to 48MHz and SPI clock to 12MHz. In this mode, CPU starts reading data from 0x000000 of SPI NOR Flash to address 0x3000 of internal SRAM of GPL162004A/GPL162005A for 256 Bytes. Thereafter, CPU will verify the SPI header tag, “0x4750”, “0x7370, 0x6973, 0x7069, 0x7370” (“GPspispisp”). Once the first 5-word tag is verified successfully, the following parameters, MCS0~MCS4 and SDRAM, will be copied to corresponding control registers and then CPU will copy data from 0x000000 of SPI NOR Flash to the designated memory address based on **DesAddrL, DesAddrH, and Sector count.** After transfer is completed, CPU will jump to the address which is the designated address plus 0x20 of offset and execute the program.

If SPI header tag check fails, CPU will set IOB4, IOB5, IOB6, and IOB7 to default settings and then turn to the SDC boot flow function immediately.

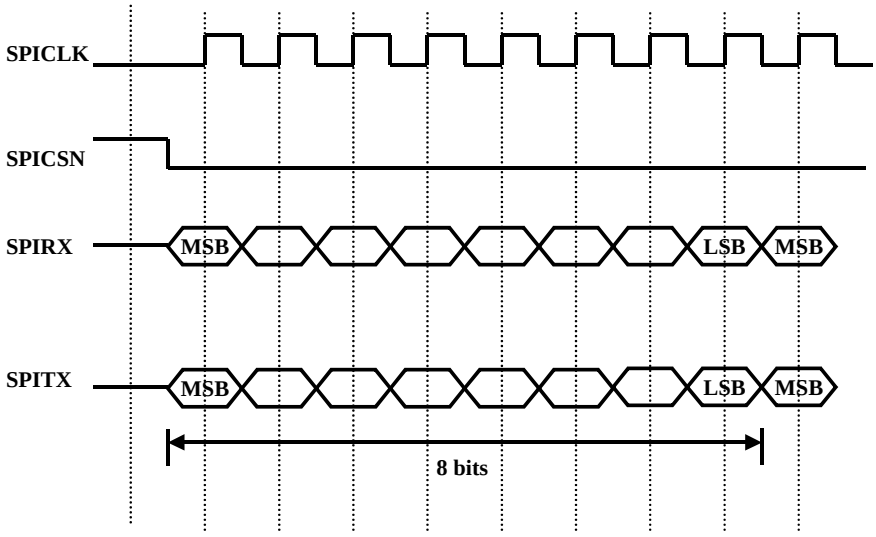
Note: Only Generalplus SPI NOR Flash is supported in GPL162004A-507A/GPL162005A-707AROM code. For more information on SPI NOR Flash, please refer to Generalplus GPR25LXXX series datasheet.

4.3.2 SPI MCU Function Code

If the 1st byte of ID is 0xD2, it means the type of SPI device is MCU; the 2nd and 3rd byte of the ID represent the SPI MCU address for storing SPI header data. GPL162004A-507A/GPL162005A-707A will fetch 256 bytes SPI MCU data from the address and store it at 0x3000 of internal RAM. The first 32-word data is a header data. Thereafter, CPU will verify the SPI header tag, "0x4750", "0x7370, 0x6973, 0x7069, 0x7370" ("GPspispisp"). Once the first 5-word tag is verified successfully, the following parameters, MCS0~MCS4, SDRAM and SPI clock, will be copied to corresponding control registers and then CPU will copy data from address **SouAddrL** and **SouAddrH** of SPI MCU to the designated memory address based on **DesAddrL, DesAddrH, and Sector count.** After transfer is done, CPU will jump to the address which is the designated address plus 0x20 of offset and execute the program. If SPI header tag check fails, SPI interface is disabled; the IOB[7..4] will return to the default status and thus, CPU clock is set to 48MHz and executes the SD card boot function..

4.4 SPI Timing Diagram

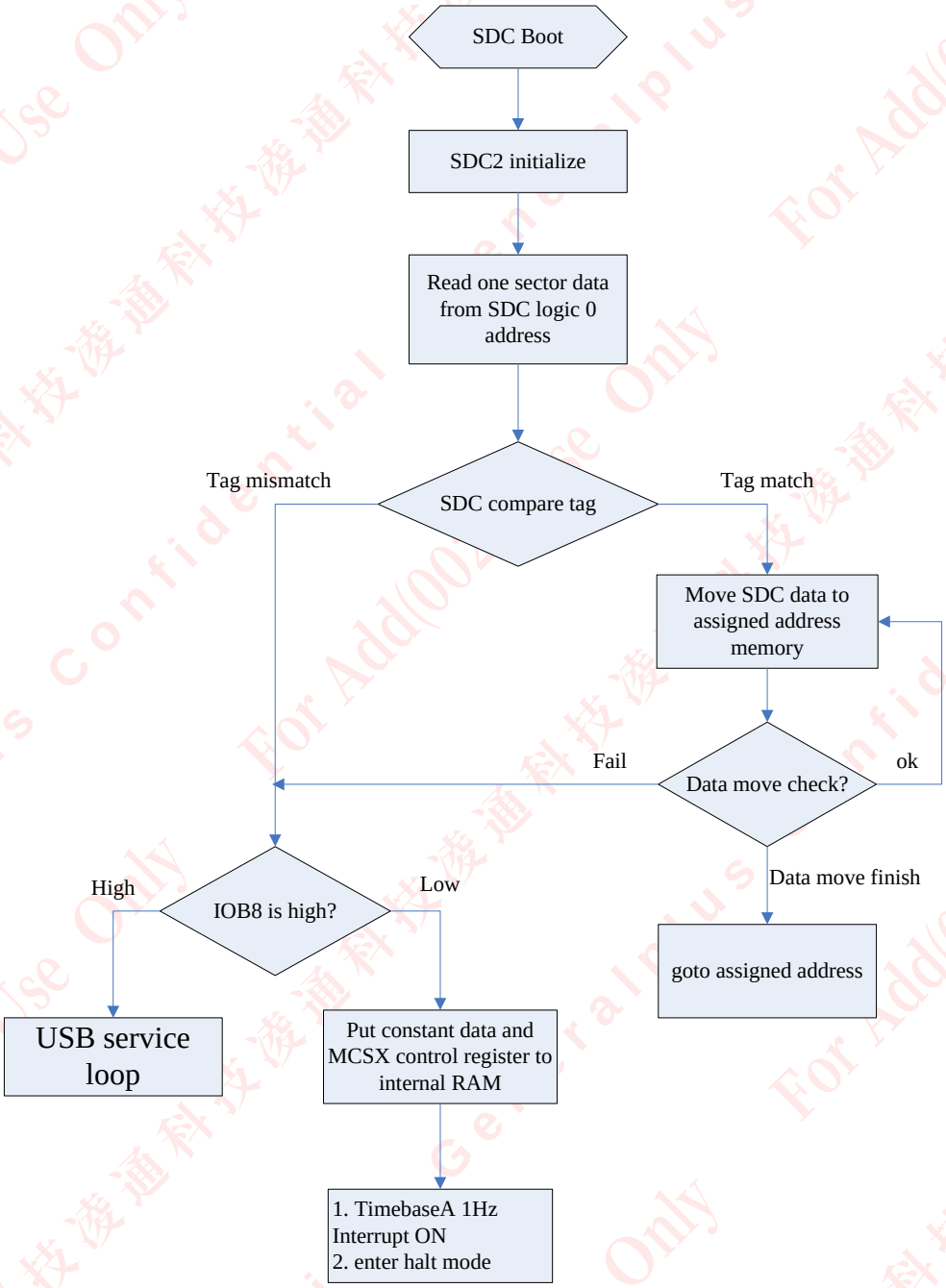
GPL162004A-507A/GPL162005A-707A SPI interface clock and data timing diagrams are as follows:



SPI control setting, Master Mode, POLARITY=0, PHASE=0

5 SDC Boot Flow

5.1 SDC Boot Flow Char



5.2 SDC Boot Header

1	2	3	4	5	6	7	8
Tag1	Tag2	Tag3	Tag4	Tag5	CS0 setup	CS1 setup	CS2 setup
9	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CS3 setup	CS4 setup	DesAddrL	DesAddrH	Sector count	PSRAM Ctrl	Write/Read Timing	0
17	**18**	**19**	**20**	**21**	**22**	**23**	**24**
SDRAM Enable	SDRAM CBRCYC	SDRAM Timing	SDRAM Ctrl1	SDRAM Misc	SDRAM Ctrl0	BCH Table AddrL	BCH Table AddrLH
25	**26**	**27**	**28**	**29**	**30**	**31**	**32**
SouAddrL	SouAddrH	Sys CLK	PLL setup	SPI CLK	0	0	0

SDC boot header includes 32 words which need to be arranged in the beginning of SDC sector 0. Afterward, user's code starts from 33rd word. The definition of each word of the header is as follows,

Tag1: 0x4750 for SDC boot

Tag2: 0X7364 for SDC boot

Tag3: 0x6373 for SDC boot

Tag4: 0x6463 for SDC boot

Tag5: 0x7364 for SDC boot

CS0 setup: the value for MCS0 control register. CPU will fill this value to the register, 0x7820, after Tag1~Tag5 are checked correctly.

CS1 setup: the value for MCS1 control register. CPU will fill this value to the register, 0x7821, after Tag1~Tag5 are checked correctly.

CS2 setup: the value for MCS2 control register. CPU will fill this value to the register, 0x7822, after Tag1~Tag5 are checked correctly.

CS3 setup: the value for MCS3 control register. CPU will fill this value to the register, 0x7823, after Tag1~Tag5 are checked correctly.

CS4 setup: the value for MCS4 control register. CPU will fill this value to the register, 0x7824, after Tag1~Tag5 are checked correctly.

DesAddrL, DesAddrH: the designated beginning memory address to store data from SDC sector 0. The memory offset from this address cannot be over 4M words of the first bank. It means that data copied from SDC can only be stored in first 4MW of memory mapping of GPL162004A/GPL162005A.

Sector count: the designated data count to be read from SDC. The unit for the data is based on sector (512 Bytes).

PSRAM Ctrl: the value for P_PSRAM_Ctrl control register. CPU will fill this value to the register, 0x7825, after Tag1~Tag5 are checked correctly.

Write/Read Timing: the value for P_RAW_WAR control register. CPU will fill this value to the register, 0x782D, after Tag1~Tag5 are checked correctly.

SDRAM Enable: the value for the control register to enable SDRAM. CPU will fill this value to the register, 0x782F, after Tag1~Tag5 are checked correctly.

SDRAM CBRCYC: the value for the control register to refresh SDRAM. CPU will fill this value to the register, 0x783D, after Tag1~Tag5 are checked correctly.

SDRAM Timing: the value for SDRAM timing control register. CPU will fill this value to the register, 0x783C, after Tag1~Tag5 are checked correctly.

SDRAM Ctrl1: the value for the control register to initialize SDRAM. CPU will fill this value to the register, 0x783B, after Tag1~Tag5 are checked correctly.

SDRAM Misc: the value for the control register which SDRAM timing adjust. CPU will fill this value to the register, 0x783E, after Tag1~Tag5 are checked correctly.

SDRAM Ctrl0: the value for the control register to initialize SDRAM. CPU will fill this value to the register, 0x783A, after Tag1~Tag5 are checked correctly.

BCH Table AddrL, BCH Table AddrH: N/A in this boot mode.

SouAddrL, SouAddrH: these two parameters are valid only on SPI MCU boot mode.

Sys CLK: N/A in this boot mode.

PLL setup: N/A in this boot mode.

SPI CLK: N/A in this boot mode.

5.3 SDC Function Code

Once SPI NOR Flash boot mode check fails, CPU will enter SD card boot flow right after. First, CPU will initialize SD card interface and then read a sector (512Byte) of data from sector 0 of a SD card to the address 0x3000 of internal RAM of GPL162004A-507A/GPL162005A-707A. After that, CPU will check the tag. If the 5-word tag is matched, CPU will configure the memory control registers, destination address and data size to copy, and SDRAM control registers based on 6^{th} ~ 22^{nd} word correspondingly. CPU will jump to (DesAddrL, DesAddrH)+0x0020 to execute after data transfer is completed. Data read from SD card is based on byte; therefore, CPU will combine 2 bytes into a word and then store to the designated address when data transfer. That is to say, the offset stored to external memory is 256 words if a sector (512 bytes) of data is read from SD card.

If SD card boot mode check fails, CPU will perform USB download check. Suppose IOB8 is detected high state, it means USB is connected and then CPU will execute USB service loop function that is able to communicate with PC and to download selected data or files to the selected on-board memory device

such as NAND Flash, NOR Flash, or SPI Flash. Hence, users should configure IOB8 as USB insertion detection signal when it is necessary to perform USB download function through internal ROM code of GPL162004A/GPL162005A.

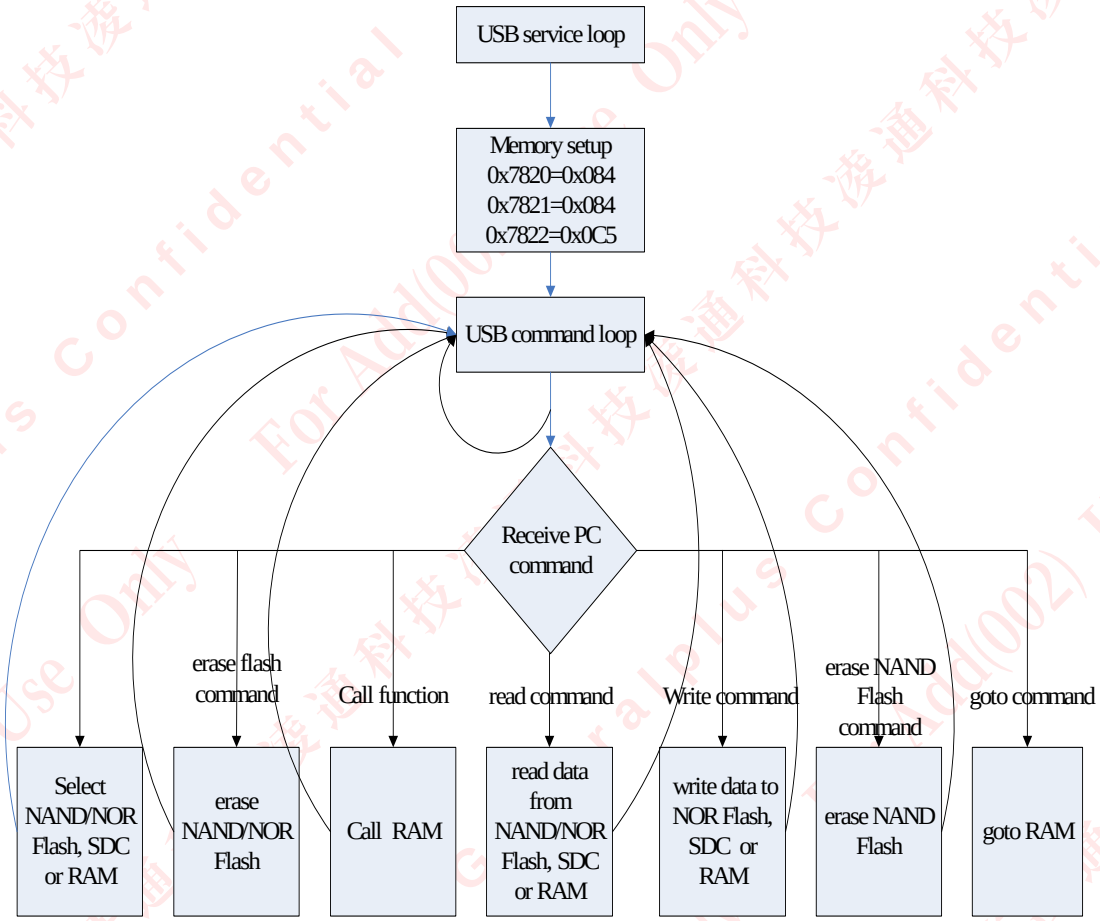
Furthermore, when IOB8 is detected low, CPU will fill the address 0x00~0x09 of internal RAM with 0x3337,0x4774, 0x5A59, 0x6824, 0x1793, and value for MCS0~MCS4 control registers and then set 1 Hz TimebaseA as interrupt/ wakeup source and finally, it enters halt mode. The above constant data is to check how CPU is awakened, either from internal ROM code of GPL162004A/GPL162005A or halt mode of user's API. On the other hand, copying the value of memory control registers to internal RAM is to protect memory settings after CPU is awakened since memory control registers will return to default after awakening from halt mode or sleep mode.

6 USB Service Loop

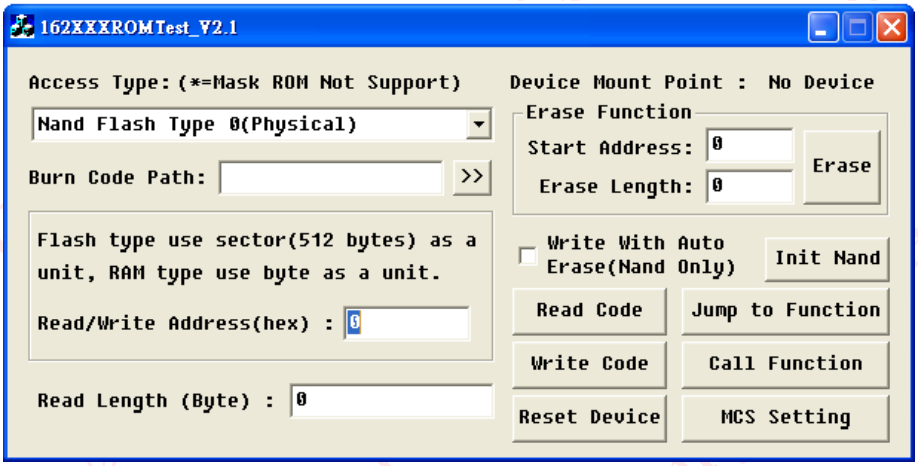
6.1 Introduction

Once IOE15 is high at CPU reset or IOB8 is detected high after CPU is awakened from internal ROM code, CPU will execute USB service loop function code automatically. The USB service function can download data from PC to on-board SPI NOR Flash, SDC or internal / external RAM of GPL162004A-507A/GPL162005A-707A through USB port in conjunction with Generalplus PC USB tools.

6.2 USB Download Flow



6.3 Tools



1. **Access Type:** select the memory type to be downloaded through GPL162004A-507A / GPL162005A-707A.

   Nand Flash Type 0(Physical): 64MB~128MB, 512+16/page NAND Flash.

   Nand Flash Type 1(Physical):128MB, 2048+64/page NAND Flash.

   Nand Flash Type 2(Physical): 256MB~32GB, 2048+64/page NAND Flash.

   RAM: Internal or external RAM.

   Nand Flash (Logical)*: Mask ROM code does not support this function, but user can download NAND Flash driver to RAM and then jump to RAM to execute NAND Flash driver. Users can use this function to perform logical read/write. This type does not support erase function.

   Nor Flash*: Mask ROM code does not support this function, but user can download NOR Flash driver to RAM and jump to RAM to execute Nor Flash driver. The user can use this function to perform NOR Flash erase/read/write function.

   SPI Flash: Only Generalplus' SPI NOR Flash is supported.

   MLC 2K page (Physical): 2KB/page type MLC NAND Flash.

   MLC 4K page(Physical): 4KB/page type MLC NAND Flash.

   SD Card: For SDC access, 007 ROM code supports logic sector read/write not through file system management.

2. **Burn Code path:** the path of the designated binary file to be loaded to NAND Flash, NOR Flash, SPI NOR Flash or SD card through GPL162004A-507A/GPL162005A-707A.
3. **Read/Write Address (hex):** The starting address to access data of GPL162004A-507A / GPL162005A-707A.
   a. If Access Type is NAND Flash type0, data access is based on a sector. In means, that sector 0 is from address 0 of page0 in the block0 of NAND0 type Flash (512 bytes of a page) and sector 1 is from address 0 of page 1 in the block 0 of NAND0 type Flash. Similarly, for NAND1 or NAND2 type Flash (2048 bytes in a page), sector 1 is from address 512 of page0 in block0. For

EEC value of each sector in writing procedure, please refer to section 3.4.

b. If Access Type is RAM, data access is based on a word. For example, 20000 of Read/Write address is loaded to 0x20000 of CS0,external RAM, and 0x0000 of Read/Write address is loaded to 0x0000 of internal RAM.

c. If access type is SPI NOR Flash, data access is based on a **word**. In read mode, reading "0000" is to read data from address 0x0000 of SPI NOR Flash and reading "0001" is to read from address 0x0001, and etc… However, writing function is based on a **sector**. For example, the address must be 0000, 0100, 0200, and 0300…, etc.; otherwise, data read from SPI NOR Flash will be shifted.

d. If Access Type is MLC 2K page, the written unit is per page (2112 bytes) and it executes NAND BCH 8-bit for ECC check.

e. If Access Type is MLC 4K page, the written unit is per page (4224 bytes) and it executes NAND BCH 8-bit for ECC check.

f. If Access Type is SD card, data access is based on a sector. For example: 0 of Read/Write address is to access SDC logic sector 0 data, and, etc..

4. **Read Length(Byte):** data length to be read from GPL162004A-507A/GPL162005A-707A by the unit of byte.
5. **Device Mount Point:** indicates GPL162004A-507A/GPL162005A-707A device mount point.
6. **Erase Function:** erases NAND/Nor Flash memory command.
   a. Start Address: Will be erased start address of NAND/Nor Flash **block address**.
   b. Erase Length: unit in block.
   c. If Access Type is RAM or SD card, this icon is not used.
7. **Write with Auto Erase:** Only valid when access type is NAND Flash 0~2. When users toggle this button, it will erase first, not write.
8. **Write Code:** write data to GPL162004A-507A/GPL162005A-707A command.
   a. When Access Type is NAND Flash Type0~2, MLC 2K page and MLC 4K page, GPL162004A-507A/GPL162005A-707A does not support these NAND Flash Types write.
   b. When Access Type is RAM, NOR Flash or SD card, data will be written to memory directly.
9. **Read Code:** Read data from GPL162004A-507A/GPL162005A-707A memory command. The Read back file name is 162XXX_readback.log.

   Jump to Function: command GPL162004A-507A/GPL162005A-707A to jump to designated address to execute. Likewise, users should set up designated address in Read/Write Address(hex).
10. **Call Function:** command GPL162004A-507A/GPL162005A-707AA to call to designated address to execute. Likewise, users should set up designated address in Read/Write Address(hex).

6.4 Application

USB download function is for two purposes. One is that it can update NAND/Nor Flash and SDC boot code and the other is to facilitate updating BIOS code or on-board mass production for customers.

6.4.1 SPI Flash and SDC boot code update

In GPL162004A-507A/GPL162005A-707A internal ROM code, writing SPI Flash or SDC, it performs logic sector access without making use of file system management or FAT. As a result, it is suitable for SDC on-board like applications; that means the SD card for boot mode on GPL162004A/GPL162005A is not for general-purpose portable mass storage applications. For instance, GPL162004A/GPL162005A can support the MLC NAND Flash application through SD card interface to access MLC NAND Flash.

6.4.2 BIOS update or Data write

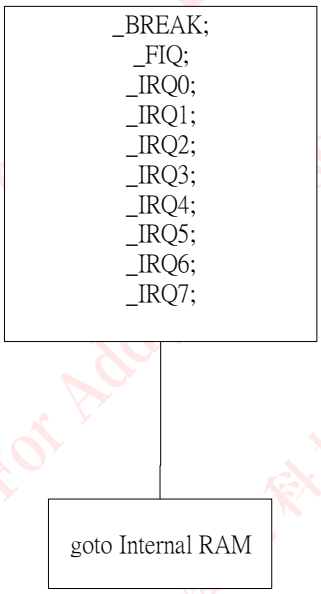
For larger BIOS code or resources data to be written to NAND Flash, it is unsuitable and risky to access NAND Flash devices directly without bad block management mechanism since some blocks other than block0 might be bad or damaged. Therefore, users should download USB function code and NAND Flash bad block management function to internal or external RAM in GPL162004A-507A / GPL162005A-707A and then execute download function in RAM. Ultimately, code/data can be downloaded accurately to good blocks of NAND type Flash through bad block management function.

Note: Generalplus provides USB function code executing in RAM to users since this code should be in accordance with USB function code of internal ROM to keep correct communication between GPL162004A-507A/GPL162005A-707A and PC.

7 ISR Vector

On GPL162004A-507A/GPL162005A-707A internal ROM, each interrupt service routine will go to corresponding designated internal address because original interrupt vectors in GPL162004A-507A/GPL162005A-707A allocated at 0xFFF5~0xFFF7 are unchangeable. Therefore, when an interrupt occurs, CPU will go to corresponding interrupt vector and execute its interrupt service routine. If GPL162004A-507A / GPL162005A-707A internal ROM code boot is applied for a project, the interrupt service routines should be programmed and allocated in designated RAM area by programmers.

The GPL162004A-507A/GPL162005A-707A internal ROM interrupt service routine flow chart is as below



The internal RAM assigned address for ISR is:

	GPL162004A/GPL162005A Internal RAM address
_BREAK	0x6FEA
_FIQ	0x6FEC
_RESET	0x6FEE
_IRQ0	0x6FF0
_IRQ1	0x6FF2
_IRQ2	0x6FF4
_IRQ3	0x6FF6
_IRQ4	0x6FF8
_IRQ5	0x6FFA
_IRQ6	0x6FFC
_IRQ7	0x6FFE

Programmers should fill data to RAM 0x6FEA~0x6FFF before using NAND/NOR Flash or SDC boot function. The data is the code of “goto address” instruction such as below:

```
goto _Ext_BREAK;        //this program used 2 word length. Put these 2 words to 0x6FEA
                        //here, assume _Ext_BREAK is _BREAK address of customer’s project

goto _Ext_IRQ0;         //this program used 2 word length. Put these 2 words to 0x6FF0
                        //here, assume _Ext_IRQ0 is _IRQ0 address of customer’s project
```

And programmers’ interrupt service routines must conform to following format:

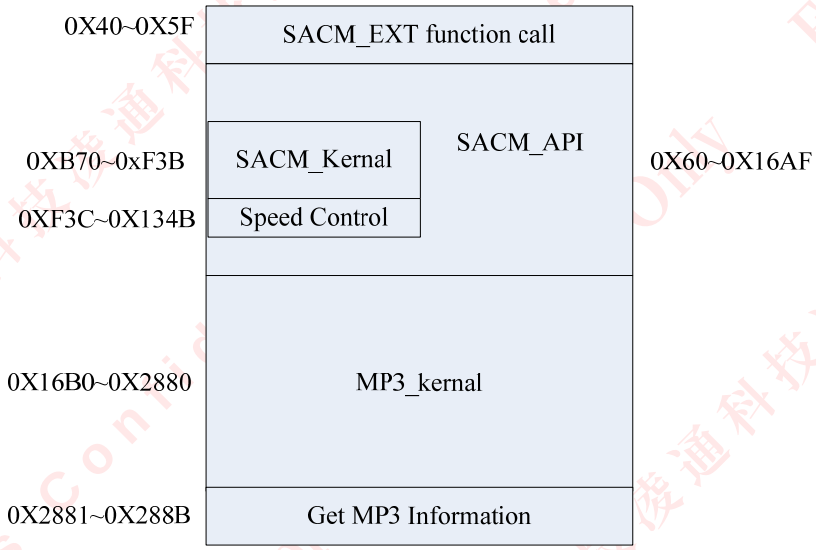
```
_Ext_Break:
        Push Rn to [sp]
        ISR service code
        …….
        Pop Rn from [sp]
        reti

_Ext_IRQ0:
        Push Rn to [sp]
        ISR service code
        …….
        Pop Rn from [sp]
        reti
```

8 Library Resource

8.1 Introduction

Generalplus has prepared some libraries for project development, including S880, DVR1800, MP3 and A128 in GPL162004A-507A/GPL162005A-707A. Generalplus will provide demo programs to customers using these libraries. The internal RAM usage on these libraries is listed as the table below.



Library type	RAM address	Size
All library used	0x40 ~ 0x288B	0x284C (10316W)
SACM_API (All speech API)	0x60 ~ 0x16AF	0x1650 (5712W)
SACM_Kernel (only S880, DVR1800)	0xB70 ~ 0xF3B	0x3CC (972W)
Speed Control	0xF3C ~ 0x134B	0x410 (1040W)
MP3 or A128 Kernel	0x16B0 ~ 0x2880	0x11D1 (4561W)
Get MP3 information code	0x2881 ~ 0x288B	0xB (11W)
BCH ECC correction code	0x2890 ~ 0x2BE0	0x351

According to the table above, performing MP3 function requires 10316 words of memory from internal RAM; however, performing S880 or DVR1800 needs 4876 (0x40 ~ 0x134B) words of memory from internal RAM. As a result, the rest of internal RAM, 0x134C ~ 0x288B, can be used by users' API.

Generalplus will provide examples of how to use API functions in your programs and some API functions are listed in the following demo program for your development reference.

8.2 Speech API

1. void SACM_Initial();

Description: Kernel initialization and calls HW initial

2. void SACM_ServiceLoop(void);

Description: decode bit stream

3. void SACM_Play(int Speech index, int Channel, int Ramp);

Description: play speech

Parameter:

Speech index: -1 : Manual mode

0 - max index : Auto mode

Channel: 1: DAC1 on

2: DAC2 on

3: DAC1, 2 on

Ramp: 0: ramp up/down off

1: ramp up on

2: ramp down on

3: ramp up/down on

4. void SACM_Stop(void);

Description: speech stop

5. void SACM_Pause(void);

Description: speech pause

6. void SACM_Resume(void);

Description: speech resume

7. void SACM_Volume(int volume);

Description: speech volume setup

Parameter:

volume: 0(mute) ~ 15(max)

8. unsigned int SACM_Status(void);

Description: get speech status

Return: C_SACM_STOP 0x8000

C_SACM_PLAY 0x0001

C_SACM_RECORD 0x0002

C_SACM_PAUSE 0x0004

9. void SACM_Speed(int speed);

Description: speech play speed setup

Parameter:

speed: 0(slowest) ~ 15(fastest)

10. void SACM_Codec(int codec);

Description: decode speech type select

Parameter:

codec: depend on how many speech used. Generalplus suggest programmers use constant value that defined in ahgorithm.h

11. void SACM_Rec(int RceMonitor,int bit_rate);

Description: speech DVR record start

Parameter:

RceMonitor: 0 off

1 on

bit_rate:

// DVR 4800 bit rate

#define BIT_RATE_32K 0

#define BIT_RATE_36K 1

#define BIT_RATE_40K 2

#define BIT_RATE_44K 3

#define BIT_RATE_48K 4

#define BIT_RATE_52K 5

#define BIT_RATE_56K 6

// DVR 1600 bit rate

#define BIT_RATE_10K 0

#define BIT_RATE_12K 1

#define BIT_RATE_14K 2

#define BIT_RATE_16K 3

#define BIT_RATE_20K 4

#define BIT_RATE_24K 5

12. void SACM_DVR1800_BITRATE(int BitRate);

Description: setup DVR1800 bit rate

Parameter:

BitRate: DVR1800 can support bit rate from 4.8K to 32K bps. From 4.8K to 12K bps, each step is 800bps. From 12K to 32K bps each step is 4K bps.

13. void SACM_MP3_SetFS(void);

Description: Get MP3 bit rate, sample rate information of MP3 source data. Before playing MP3, programmers should call this function first. The MP3 bit rate and sample rate will put in two variable, named MP3_BR and MP3_FS.

14. void USER_SetStartAddr(int index);

Description: Speech source data address defined by users.

Parameter: index: speech index.

15. void USER_SetRECStartAddr(long address);

Description: DVR encoded data store address defined by users.

Parameter: address: encoded data store address.

8.3 Call Back Function

The demo program shows some function addresses need to be placed from 0x40 in internal RAM.

```
R1 = 0x40;
R2 = seg F_SACM_Stop;                          //sacm_stop address
R2 += 0xFE80;
[R1++] = R2;
R2 = offset F_SACM_Stop;
[R1++] = R2;

[R1++] = R2;                                   //reserved
[R1++] = R2;

R2 = seg F_SACM_MP3_System_Get_BS_Manual;      //MP3 bit stream get function code address
R2 += 0xFE80;
[R1++] = R2;
R2 = offset F_SACM_MP3_System_Get_BS_Manual;
[R1++] = R2;
```

```
R2 = seg F_SACM_Decode_Process;                    //sacm_decode function address
R2 += 0xFE80;
[R1++] = R2;
R2 = offset F_SACM_Decode_Process;
[R1++] = R2;
```